Technical Communications of the International Conference on Logic Programming, 2010 (Edinburgh), pp. 236–240
http://www.floc-conference.org/ICLP-home.html

BISIMILARITY IN CONCURRENT CONSTRAINT PROGRAMMING

ANDRÉS A. ARISTIZÁBAL P.

CNRS, LIX École Polytechnique and INRIA Team COMÈTE
Route de Saclay 91128 Palaiseu Cedex, France.
E-mail address: andresaristi@lix.polytechnique.fr
URL: http://www.lix.polytechnique.fr/∼andresaristi/

Abstract. In this doctoral work we aim at developing a new approach to labelled semantics and equivalences for the Concurrent Constraint Programming (CCP) which will enable a broader capture of processes behavioural equivalence. Moreover, we work towards exploiting the strong connection between first order logic and CCP. Something which will allow us to represent logical formulae in terms of CCP processes and verify its logical equivalence by means of our notion of bisimilarity. Finally, following the lines of the Concurrecy Workbench we plan to implement a CCP Workbench based on our theoretical structure.

Motivations

Concurrency is concerned with the fundamental aspects of systems consisting of multiple computing agents, usually called *processes*, that interact among each other. *Bisimilarity* is a central behavioural equivalence in concurrency theory as it elegantly captures our intuitive notion of process equivalence; two processes are equivalent if they can match each other's moves. In fact, several concurrent formalisms such as CCS [Mil80] and the π-calculus [Mil99] are equipped with semantic, axiomatic, verification and, in general, reasoning techniques for bisimilarity.

Concurrent Constraint Programming (CCP) [Sar90] is a well-established *declarative* formalism for concurrency. Its basic intuitions arise mostly from *first-order logic*. In CCP processes can interact by *adding* (or *telling*) partial information in a medium, a so-called *store*. Partial information is represented by constraints (e.g., $x > 42$) on the shared variables of the system. The other way in which processes can interact is by *asking* partial information to the store. This provides the synchronization mechanism of the model; asking agents are suspended until there is enough information in the store to answer their query.

Despite the relevance of bisimilarity on the behavioural theory of processes, there have been few attempts to define a proper notion of bisimilarity equivalence for CCP. Apart from the rich reasoning techniques that are typically derived from this equivalence, the close ties between CCP and logic may provide with a novel characterization of logic equivalence in terms of bisimilarity.

1998 ACM Subject Classification: D.1.3, D.3.2, D.3.3, F.1.1, F.1.2, F.3.2, F.4.0.
Key words and phrases: Concurrent Constraint Programming, Concurrency, Behavioural Equivalence, Bisimilarity, Process Calculi, Operational Semantics, Labelled Semantics.

Technical Communications of the 26th International Conference on Logic Programming, Edinburgh, July, 2010
Editors: Manuel Hermenegildo, Torsten Schaub
LIPIcs - Leibniz International Proceedings in Informatics. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany
Digital Object Identifier: 10.4230/LIPIcs.ICLP.2010.236

1. Goals

We aim to provide CCP with an appropriate notion of bisimilarity and its derived reasoning techniques. Furthermore, we plan to use the close connection between CCP and first order logic to give a characterization of logical equivalence in terms of bisimilarity. Finally, we plan to implement an automated tool for verifying bisimilarity equivalence of CCP processes along the lines of the Concurrency Workbench [Cle93] .

2. Current Work

Like in other process algebras, in CCP processes are represented as syntactic terms reflecting their structure. For example, $tell(c)$ represents the process that adds the constraint c to the store and $ask(c).P$ is a process that asks if c can be derived from the information in the store and if so, it executes the process P. The composite term $P \parallel Q$ represents the execution of the processes P and Q in parallel.

In [Sar91] the authors gave an operational semantics for CCP which we will refer to as reduction semantics. Intuitively, a reduction $\langle P, S\rangle \rightarrow \langle P', S'\rangle$ represents a one-step evolution of the process-store configuration $\langle P, S\rangle$ to $\langle P', S'\rangle$.

We use $\models$ to denote an entailment relation specifying interdependencies between constraints (e.g. $x > 10 \models x > 5$). We follow the well-established notion of barbed bisimilarity for the π-calculus [Mil99] and introduce the corresponding notion for CCP:

Definition 2.1. *(Barbed bisimilarity)* A barbed bisimulation is a symmetric relation $\mathcal{R}$ s.t., $\langle P, S_p\rangle \mathcal{R} \langle Q, S_q\rangle$ implies that:

(i) if $\langle P, S_p\rangle \rightarrow \langle P', S'_p\rangle$ then $\exists \langle Q', S'_q\rangle : \langle Q, S_q\rangle \rightarrow \langle Q', S'_q\rangle$ and $\langle P', S'_p\rangle \mathcal{R} \langle Q', S'_q\rangle$, and
(ii) $S_p \models S_q$.

We say that $\langle P, S_p\rangle$ and $\langle Q, S_q\rangle$ are barbed bisimilar, written $\langle P, S_p\rangle \dot{\sim}_B \langle Q, S_q\rangle$, if there is a barbed bisimulation $\mathcal{R}$ s.t. $\langle P, S_p\rangle \mathcal{R} \langle Q, S_q\rangle$.

Unfortunately, there are barbed bisimilar processes that when placed in a given context are not longer equivalent. Roughly, a context $C[\cdot]$ is a process term with a single hole $\cdot$ such that replacing $\cdot$ with a process gives a well-formed process. E.g., by taking $P = ask(x > 0).tell(y = 0)$ and $Q = ask(x > 10).tell(y = 1)$ and $C[\cdot] = tell(x > 5) \parallel \cdot$ we can verify that $P \dot{\sim}_B Q$ but $C[P] \not\dot{\sim}_B C[Q]$. Thus, we define:

Definition 2.2. *(Barbed Congruence)* We say that P and Q are barbed congruent, written $P \sim_B Q$, if for all contexts $C[\cdot]$, $\langle C[P], true\rangle \dot{\sim}_B \langle C[Q], true\rangle$.

The above definition is rather unsatisfactory because of the quantification over all possible contexts. To deal with this we define a labelled transition semantics. Intuitively, a transition $\langle P, S\rangle \xrightarrow{\alpha} \langle P', S'\rangle$ labelled with a constraint α, represents the minimal constraint α that needs to be added to the store S to evolve from $\langle P, S\rangle$ into $\langle P', S'\rangle$.

Our work builds on a similar CCP labelled semantics introduced in [Sar90]. The notion of bisimilarity in [Sar90] is, however, over-discriminating; e.g., it distinguishes $P = ask(x < 10).tell(y = 0) \parallel ask(x < 10).tell(y = 0)$ from $Q = ask(x < 5).tell(y = 0) \parallel ask(x < 10).tell(y = 0)$ which are clearly equivalent. Our notion of bisimilarity is defined thus:

Definition 2.3. *(Strong bisimilarity)* A strong bisimulation is a symmetric relation $\mathcal{R}$ s.t., $\langle P, S_p\rangle \mathcal{R} \langle Q, S_q\rangle$ implies that:

(i) if $\langle P, S_p\rangle \xrightarrow{\alpha} \langle P', S'_p\rangle$ then $\exists\langle Q', S'_q\rangle : \langle Q, S_q \wedge \alpha\rangle \rightarrow \langle Q', S'_q\rangle$ and $\langle P', S'_p\rangle \mathcal{R} \langle Q', S'_q\rangle$ and
(ii) $S_p \models S_q$.

We say that $\langle P, S_p\rangle$ and $\langle Q, S_q\rangle$ are strong bisimilar, written $\langle P, S_p\rangle \dot{\sim} \langle Q, S_q\rangle$, if there exists a strong bisimulation $\mathcal{R}$ such that $\langle P, S_p\rangle \mathcal{R} \langle Q, S_q\rangle$.

The main result we have obtained so far that the above notion fully captures barbed congruence but without quantification over all possible contexts: I.e., we state:

Theorem 2.4. *$\langle P, S_p\rangle \dot{\sim} \langle Q, S_q\rangle$ if and only if $\langle P, S_p\rangle \sim_B \langle Q, S_q\rangle$.*

Acknowledgement

This work is supervised by Catuscia Palamidessi and Frank Valencia in collaboration with Filippo Bonchi in the context of the INRIA project FORCES.

References

[Bon08] Filippo Bonchi. Abstract semantics by observable contexts. In *ICGT '08: Proceedings of the 4th international conference on Graph Transformations*, pp. 478–480. Springer-Verlag, Berlin, Heidelberg, 2008. doi:http://dx.doi.org/10.1007/978-3-540-87405-8_38.

[Cle93] Rance Cleaveland, Joachim Parrow, and Bernhard Steffen. The concurrency workbench: A semantics-based tool for the verification of concurrent systems. *ACM Trans. Program. Lang. Syst.*, 15(1):36–72, 1993.

[Mil80] Robin Milner. *A Calculus of Communicating Systems*, *Lecture Notes in Computer Science*, vol. 92. Springer, 1980.

[Mil99] Robin Milner. *Communicating and mobile systems: the π-calculus.* Cambridge University Press, New York, NY, USA, 1999.

[Sar90] Vijay A. Saraswat and Martin C. Rinard. Concurrent constraint programming. In *POPL*, pp. 232–245. 1990.

[Sar91] Vijay A. Saraswat, Martin C. Rinard, and Prakash Panangaden. Semantic foundations of concurrent constraint programming. In *POPL*, pp. 333–352. 1991.

Appendix A. Proof of Theroem 2.4

Firstly we show out new definitions and lemmas to proof our main theorem.

Another alternative definition for the barbed congruence is what we will name as a saturated barbed bisimilarity. This will be rather important since its definition is a bit more specific towards CCP than a barbed congruece, therefore is easier to relate with the strong bisimilarity we will define later on.

Definition A.1. *(Saturated barbed bisimilarity).* A saturated barbed bisimulation is a symmetric binary relation $\mathcal{R}$ on tuples of processes and stores satisfying the following: $\langle P, S_p\rangle \mathcal{R} \langle Q, S_q\rangle$ implies that:

(i) if $\langle P, S_p\rangle \to \langle P', S'_p\rangle$ then $\exists \langle Q', S'_q\rangle : \langle Q, S_q\rangle \to \langle Q', S'_q\rangle$ and $\langle P', S'_p\rangle \mathcal{R} \langle Q', S'_q\rangle$.
(ii) $S_p \models S_q$.
(iii) $\forall S' \langle P, S_p \wedge S'\rangle \mathcal{R} \langle Q, S_q \wedge S'\rangle$.

We say that $\langle P, S_p\rangle$ and $\langle Q, S_q\rangle$ are saturated barbed bisimilar, written $\langle P, S_p\rangle \dot{\sim}_{SB} \langle Q, S_q\rangle$, if there exists a saturated barbed bisimulation $\mathcal{R}$ such that $\langle P, S_p\rangle \mathcal{R} \langle Q, S_q\rangle$.

We did not report neither the reduction semantics nor the labelled semantics for lack of space. In order to prove our main theorem we assume that the two following lemmas hold.

Lemma A.2. *(Soundness of labelled semantics). If* $\langle P, S_p\rangle \xrightarrow{\alpha} \langle P', S'_p\rangle$*, then* $\langle P, S_p \wedge \alpha\rangle \to \langle P', S'_p\rangle$*.*

Lemma A.3. *(Completeness of labelled semantics). If* $\langle P, S_p \wedge x\rangle \to \langle P', S'_p\rangle$ *then* $\exists y, z$ *s.t.* $\langle P, S_p\rangle \xrightarrow{y} \langle P', S''_p\rangle$ *and* $(y \wedge z = x) \wedge (S''_p \wedge z = S'_p)$*.*

Corollary A.4. $\langle P, S_p\rangle \xrightarrow{true} \langle P', S'_p\rangle$ *if and only if* $\langle P, S_p\rangle \to \langle P', S'_p\rangle$

Theorem A.5. $\langle P, S_p\rangle \dot{\sim} \langle Q, S_q\rangle \Rightarrow \forall S' \langle P, S_p \wedge S'\rangle \dot{\sim} \langle Q, S_q \wedge S'\rangle$

Proof. We take a strong bisimulation $\mathcal{R} = \{(\langle P, S_p \wedge S'\rangle, \langle Q, S_q \wedge S'\rangle)$ s.t. $\langle P, S_p\rangle \dot{\sim} \langle Q, S_q\rangle\}$

(i) $\langle P, S_p \wedge S'\rangle \xrightarrow{\alpha} \langle P', S'_p\rangle$
By Lemma A.2 $\langle P, S_p \wedge S' \wedge \alpha\rangle \to \langle P', S'_p\rangle$.
By Lemma A.3 $\langle P, S_p\rangle \xrightarrow{y} \langle P', S''_p\rangle$ and $(y \wedge z = S' \wedge \alpha) \wedge (S''_p \wedge z = S'_p)$. Since $\langle P, S_p\rangle \dot{\sim} \langle Q, S_q\rangle$, then $\langle Q, S_q \wedge y\rangle \to \langle Q', S''_q\rangle$ s.t. $\langle P', S''_p\rangle \dot{\sim} \langle Q', S''_q\rangle$. Note that all reductions are preserved when adding constraints to the store, therefore from $\langle Q, S_q \wedge y\rangle \to \langle Q', S''_q\rangle$ we can derive that $\langle Q, S_q \wedge y \wedge z\rangle \to \langle Q', S''_q \wedge z\rangle$. This means that $\langle Q, S_q \wedge S' \wedge \alpha\rangle \to \langle Q', S''_q \wedge z\rangle$. Now we have that $\langle P', S'_p\rangle = \langle P', S''_p \wedge z\rangle \mathcal{R} \langle Q', S''_q \wedge z\rangle$, because $\langle P', S''_p\rangle \dot{\sim} \langle Q', S''_q\rangle$.
(ii) $S_p \wedge S' \models S_q \wedge S'$ since $S_p \models S_q$ by $\langle P, S_p\rangle \dot{\sim} \langle Q, S_q\rangle$ and $S' = S'$.

■

Now we state the lemmas which will enable us to prove our main theorem.

Lemma A.6. $\langle P, S_p\rangle \dot{\sim} \langle Q, S_q\rangle \Rightarrow \langle P, S_p\rangle \dot{\sim}_{SB} \langle Q, S_q\rangle$.

Proof. There exists a saturated barbed bisimulation $\mathcal{S}$ s.t. $\mathcal{S} = \{(\langle P, S_p\rangle, \langle Q, S_q\rangle)$ s.t. $\langle P, S_p\rangle \dot{\sim} \langle Q, S_q\rangle\}$ if the following conditions are fulfilled:

(i) if $\langle P, S_p\rangle \rightarrow \langle P', S'_p\rangle$ then $\exists\langle Q', S'_q\rangle : \langle Q, S_q\rangle \rightarrow \langle Q', S'_q\rangle$ and $\langle P', S'_p\rangle \; \mathcal{S} \; \langle Q', S'_q\rangle$. Suppose that $\langle P, S_p\rangle \rightarrow \langle P', S'_p\rangle$ then by Corollary A.4 $\langle P, S_p\rangle \xrightarrow{true} \langle P', S'_p\rangle$. Since $\langle P, S_p\rangle \dot{\sim} \langle Q, S_q\rangle$ then $\langle Q, S_q \wedge true\rangle \rightarrow \langle Q', S'_q\rangle$ then $\langle Q, S_q\rangle \rightarrow \langle Q', S'_q\rangle$ and $\langle P, S_p\rangle \dot{\sim} \langle Q, S_q\rangle$ then $\langle P, S_p\rangle \; \mathcal{S} \; \langle Q, S_q\rangle$

(ii) $S_p \models S_q$. Since $P \dot{\sim} Q$ (Condition (ii)).

(iii) $\forall S' \langle P, S_p \wedge S'\rangle \; \mathcal{R} \; \langle Q, S_q \wedge S'\rangle$. By Theorem A.5

■

Lemma A.7. $\langle P, S_p\rangle \dot{\sim}_{SB} \langle Q, S_q\rangle \Rightarrow \langle P, S_p\rangle \dot{\sim} \langle Q, S_q\rangle$.

Proof. There exists a strong bisimulation $\mathcal{R}$ s.t. $\mathcal{R} = \{(\langle P, S_p\rangle, \langle Q, S_q\rangle) \text{ s.t. } \langle P, S_p\rangle \dot{\sim}_{SB} \langle Q, S_q\rangle\}$ and if the following conditions are fulfilled:

(i) if $\langle P, S_p\rangle \xrightarrow{\alpha} \langle P', S''_p\rangle$ then $\exists\langle Q', S''_q\rangle : \langle Q, S_q \wedge \alpha\rangle \rightarrow \langle Q', S''_q\rangle$ and $\langle P', S''_p\rangle \; \mathcal{R} \; \langle Q', S''_q\rangle$. Suppose that $\langle P, S_p\rangle \xrightarrow{\alpha} \langle P', S'_p\rangle$ then by Lemma A.2 $\langle P, S_p \wedge \alpha\rangle \rightarrow \langle P', S'_p\rangle$. Since $\langle P, S_p\rangle \dot{\sim}_{SB} \langle Q, S_q\rangle$ then $\langle Q, S_q \wedge \alpha\rangle \rightarrow \langle Q', S'_q\rangle$ s.t. $\langle P', S'_p\rangle \dot{\sim}_{SB} \langle Q', S'_q\rangle$ then $\langle P', S'_p\rangle \; \mathcal{R} \; \langle Q', S'_q\rangle$

(ii) $S_p \models S_q$. Since $P \dot{\sim}_{SB} Q$ (Condition (ii)).

■

Theorem 2.4

Proof. By Lemma A.6 and Lemma A.7. ■